

PIPELINED MAXIMAL-SIZED MATCHING CELL DISPATCH SCHEDULING

§ 0. PRIORITY CLAIM

5 Benefit is claimed, under 35 U.S.C. § 119(e)(1),
to the filing date of: provisional patent application
serial number __/____,____, entitled "PMM: A PIPELINED
MAXIMAL-SIZED MATCHING SCHEDULING APPROACH FOR
INPUT-BUFFERED SWITCHES", filed on May 31, 2001 and listing
10 Eiji Oki, Roberto Rojas-Cessa and Jonathan Chao as the
inventors, for any inventions disclosed in the manner
provided by 35 U.S.C. § 112, ¶ 1. This provisional
application is expressly incorporated herein by reference.

15 § 1. BACKGROUND

§ 1.1 FIELD OF THE INVENTION

20 The present invention concerns the communication
of data over networks, such as the Internet for example.
More specifically, the present invention concerns
scheduling the servicing (e.g., dispatching) of cells or
packets buffered at input ports of a switch.

25 § 1.2 RELATED ART

30 Switches and routers are used in networks, such as the
Internet for example, to forward data towards its destination.
The need for high-speed switches and routers is introduced in
§ 1.2.1 below. Then, input buffering, as used in high-speed
switches, is introduced in § 1.2.2 below.

§ 1.2.1 THE NEED FOR LARGE-SCALE AND HIGH-SPEED (e.g., TERABIT) ROUTERS AND SWITCHES

Many expect that Internet traffic will continue to grow explosively. Given this assumption, it is expected that high-speed switches and routers (e.g., those having a throughput over one Terabit per second) will become necessary. Most high-speed packet switches adopt a fixed-size cell in the switch fabric. If variable length packets are to be supported in the network, such packets may be segmented and/or padded into fixed-sized cells upon arrival, switched through the fabric of the switch, and reassembled into packets before departure. Input buffering is introduced below in § 1.2.2 as a way to handle these incoming cells.

§ 1.2.2 BUFFERING IN HIGH-SPEED ROUTERS AND SWITCHES

There are various types of buffering strategies in switch architectures: input buffering, output buffering, or crosspoint buffering. Information on these strategies can be found in the following articles: G. Nong and M. Hamdi, "On the Provision of Quality-of-Service Guarantees for Input Queued Switches," IEEE Commun. Mag., Vol. 38, No. 12, pp. 62-69 (Dec. 2000); E. Oki, N. Yamanaka, Y. Ohtomo, K. Okazaki, and R. Kawano, "A 10-Gb/s (1.25 Gb/s \times 8) 4 \times 2 0.25-micrometer CMOS/SIMOX ATM Switch Based on Scalable Distribution Arbitration," IEEE J. Solid-State Circuits, Vol. 34, No. 12, pp. 1921-1934 (Dec. 1999); and J. Turner and N. Yamanaka, "Architectural Choices in Large Scale ATM Switches," IEICE Trans. Commun., Vol. E81-B, No. 2, pp. 120-137 (Feb. 1998). Each of these articles is incorporated herein by reference. Input buffering is a cost effective approach for high-speed

switches. This is because input-buffered switches do not require internal speedup, nor do they allocate buffers at each crosspoint. They also relax memory-bandwidth and memory-size constraints.

5

§ 1.2.2.1 THE USE OF VIRTUAL OUTPUT QUEUES TO AVOID HEAD-OF-LINE BLOCKING

It is well known that head-of-line ("HOL") blocking
 10 limits the maximum throughput (e.g., to 58.6%) in an
 input-buffered switch with a First-In-First-Out (FIFO)
 structure. See, e.g., the article, M. J. Karol, M. G. Hluchyj,
 and S. P. Morgan, "Input Versus Output Queuing on a
 Space-Division Packet Switch," IEEE Trans. Commun., Vol. COM-35,
 15 pp. 1347-1356 (1987). This article is incorporated herein by
 reference. The article, N. McKeown, "The iSLIP Scheduling
 Algorithm for Input-Queued Switches," IEEE/ACM Trans.
Networking, Vol. 7, No. 2, pp. 188-200 (Apr. 1999), shows using
 a Virtual-Output-Queue (VOQ) structure to overcome HOL-blocking.
 20 This article is incorporated herein by reference.

In an input-buffered switch that uses VOQs, a
 fixed-size cell is sent from any input to any output, provided
 that, in a given time slot, no more than one cell is sent from
 25 the same input, and no more than one cell is received by the
 same output. Each input port has N VOQs, one for each of N
 output ports. The HOL cell in each VOQ can be selected for
 transmission across the switch in each time slot. Therefore,
 every time slot, a scheduler has to determine one set of
 30 matching. That is, for each of the output ports, the scheduler
 may match one of the corresponding VOQs with the output port.

§ 1.2.2.2 MAXIMUM-SIZED AND MAXIMAL-SIZED MATCHING ALGORITHMS IN HIGH SPEED SWITCHES

Maximum-sized matching algorithms to schedule the input-output matching for input-buffered switches with VOQs, that achieve 100% throughput have been proposed. See, e.g., the articles: J. E. Hopcroft and R. M. Karp, "An Algorithm for Maximum Matching in Bipartite Graphs," Soc. Ind. Appl. Math J. Computation, Vol. 2, pp. 225-231 (1973); and N. McKeon, A. Mekittikul, V. Anantharam, and J. Walrand, "Achieving 100% Throughput in Input-Queued Switches," IEEE Trans. Commun., Vol. 47, No. 8, pp. 1260-1267 (Aug. 1999). These articles are incorporated herein by reference. Unfortunately, these algorithms are hard to implement in high-speed switches because of their high computing time complexity.

Maximal-sized matching algorithms have been proposed as an alternative to the maximum-sized matching ones. Two of these algorithms, iSLIP and Dual Round-Robin Matching (DRRM), are described in the articles: N. McKeown, "The iSLIP Scheduling Algorithm for Input-Queued Switches," IEEE/ACM Trans. Networking, Vol. 7, No. 2, pp. 188-200 (Apr. 1999); H. J. Chao and J. S. Park, "Centralized Contention Resolution Schemes for a Large-Capacity Optical ATM Switch," Proc. IEEE ATM Workshop '97, Fairfax, VA (May 1998); and H. J. Chao, "Saturn: A Terabit Packet Switch Using Dual Round-Robin," IEEE Commun. Mag., Vol. 38, No. 12, pp. 78-84 (Dec. 2000). These articles are incorporated herein by reference. The computing complexity of the iSLIP and DRRM methods are less than maximum matching methods. Moreover, the iSLIP and DRRM methods provide 100% throughput under uniform traffic and complete fairness for best-effort traffic. However, in each of these methods, the

maximal matching is to be completed within one cell time slot. Such a constraint may become unacceptable as the switch size increases and/or the port speed becomes high, because the arbitration time becomes longer than one time slot or the time slot shrinks, respectively. For example, for a 64-byte fixed-length cell at a port speed of 40Gbit/s (OC-768), the computation time for completing maximal-sized matching is only 12.8 ns. Existing proposals for relaxing the time constraints are discussed below in § 1.2.2.3.

§ 1.2.2.3 ROUND-ROBIN GREEDY SCHEDULING (RRGS)

To relax the scheduling timing constraint, a pipelined-based scheduling algorithm called Round-Robin Greedy Scheduling (RRGS) is proposed in the article: A. Smiljanic, R. Fan, and G. Ramamurthy, "RRGS-Round-Robin Greedy Scheduling for Electronic/Optical Terabit Switches," Proc. IEEE Globecom '99, pp. 1244-1250 (1999). This article is incorporated herein by reference. With RRGS, each input has only to perform one round-robin arbitration within one time slot to select one VOQ. However, if a given switch has N inputs, then N input round-robin operations (that select its cell to be transmitted at a given time slot T) are allocated into the different previous N time slots $\{T-N, T-N+1, \dots, T-1\}$ in a simple cyclic manner so that RRGS can avoid output contention.

Unfortunately, RRGS can't provide max-min fair share for a best-effort service. For example, let $\lambda(i,j)$ be the input offered load to VOQ(i,j) and let $\mu(i,j)$ be the acceptable transmission rate from VOQ(i,j). Consider a 3×3 switch in which $\lambda(0,0) = \lambda(1,0) = 1.0$ and in which other input offered

loads $\lambda(i,j) = 0$. According to the RRGs algorithm, the acceptable transmission rate is obtained as $\mu(0,0) = 2/3$ and $\mu(1,0) = 1/3$. Thus, when traffic is not balanced, some inputs can unfairly send more cells than others. The article, A.

5 Similjanic, "Flexible Bandwidth Allocation in Terabit Packet Switches," Proc. IEEE Workshop on High Performance Switching and Routing 2000, pp. 233-239 (2000) proposes weighted-RRGS ("WRRGS"), which guarantees pre-reserved bandwidth. This article is incorporated herein by reference. However, even with
10 WRRGS, fairness is not guaranteed for best-effort traffic. In addition, once every N time-slot cycles, an idle time slot is produced when N is an even number. This means that RRGs does not maximize its use of the switching capacity.

15 § 1.2.3 UNMET NEEDS

In view of the foregoing limits of proposed maximal matching scheduling schemes, one that (i) relaxes the scheduling time into more than one time slot, (ii)
20 provides high throughput, and/or (iii) maintains fairness for best-effort traffic, is still desired.

§ 2. SUMMARY OF THE INVENTION

25 The present invention may be used to meet the unmet needs outlined in § 1.2.3 above. The present invention may do so by operating, in a pipelined manner, several subschedulers within a main scheduler. This permits each subscheduler to take more than one time slot
30 to complete its matching. For each time slot, one of the subschedulers provides the matching result. Since the subschedulers collectively generate one match per time

slot, the subschedulers may adopt a pre-existing maximal matching algorithm, such as iSLIP or DRRM for example, thereby taking advantage of the desirable properties of these algorithms -- namely 100% throughput under uniform traffic and fairness for best-effort traffic -- found in their non-pipelined versions.

Assuming that it takes each subscheduler K (or between K-1 and K) time slots to perform the matching, in one embodiment, a main scheduler may have K subschedulers. Assuming that cells arrive at the switch in a uniform fashion, each subscheduler may begin arbitration in successive time slots. Since there are K subschedulers, each one may use K time slots to perform a match. Accordingly, the main scheduler may produce a match every time slot.

§ 3. BRIEF DESCRIPTION OF THE DRAWINGS

Figure 1 is a block diagram of an exemplary input buffered switch with which the present invention may be used.

Figure 2 is a bubble chart illustrating operations that may be performed by the present invention.

Figure 3 is a flow diagram illustrating an exemplary virtual output queue method that may be performed under the present invention.

Figures 4A and 4B are flow diagrams illustrating exemplary main scheduler methods that may be performed

under the present invention.

Figure 5 is a flow diagram illustrating an exemplary matching method that may be performed under the present invention.

Figure 6 is a block diagram of an exemplary apparatus, as well as inter-component signaling links, that may be used to perform operations under the present invention.

Figures 7(a) through 7(c) provide an example that illustrates an operation of an exemplary pipelined maximal-sized matching scheduling approach under the present invention.

Figure 8 is a timing diagram illustrating the operation of subschedulers in the example of Figures 7(a) through 7(c).

§ 4. DETAILED DESCRIPTION OF THE INVENTION

The present invention involves methods, and apparatus for scheduling the dispatch of cells or packets in a high-speed input buffered switch. The following description is presented to enable one skilled in the art to make and use the invention, and is provided in the context of particular embodiments and methods. Various modifications to the disclosed embodiments and methods will be apparent to those skilled in the art, and the general principles set forth below may be applied to other embodiments, methods and applications. Thus, the present

invention is not intended to be limited to the embodiments and methods shown and the inventors regard their invention as the following disclosed methods, apparatus and materials and any other patentable subject matter to the extent that
5 they are patentable.

§ 4.1 EXEMPLARY ENVIRONMENT

The present invention may be used with an
10 input-buffered switch with virtual output queues (VOQs). For example, as shown in Figure 1, the switch 100 may include input (ports) 110, a main scheduler 120 and (e.g., crossbar) switching fabric 130. As shown in Figure 1, there may be N inputs. In one embodiment, these N inputs
15 may be grouped across m input modules (not shown). Each input 110 will preferably have a number of VOQs corresponding to the number of outputs. Thus, if there are N outputs, each input will have N VOQs.

20 In an exemplary embodiment of the invention, the main scheduler 120 may include N^2 request counters, and K subschedulers. Each of the K subschedulers may have N^2 request flags. Each of the K subschedulers may operate in a pipeline manner, thus allowing each subscheduler to take
25 K time slots, and still produce a match (using, e.g., DRRM, iSLIP, or another maximal matching technique) in every cell time slot.

In the explanation of an exemplary embodiment of
30 the invention, the following terms will be used:

N = the number of input ports and output ports;

$K \equiv$ the number of subschedulers in each input
module;

$VOQ(i,j) \equiv$ the VOQ of input i that stores cells
destined for output j , where $0 \leq i \leq N-1$,
5 and $0 \leq j \leq N-1$;

$RC(i,j) \equiv$ the request counter associated with $VOQ(i,j)$;

$L_{\max} \equiv$ the maximum VOQ occupancy;

$C(i,j) \equiv$ the value of $RC(i,j)$ -- the number of
accumulated requests associated with $VOQ(i,j)$
10 that have not been sent to any subscheduler,
where $0 \leq C(i,j) \leq L_{\max}$;

$RF(i,j,k) \equiv$ the request flag associated with $VOQ(i,j)$
and subscheduler k , where $0 \leq k \leq K-1$;

$F(i,j,k) \equiv$ the value of $RF(i,j,k)$, where $0 \leq F(i,j,k)$
15 ≤ 1 ;

When $F(i,j,k)$ is set to one, this indicates that input i
has a request to output j in subscheduler k . When $F(i,j,k)$
is set to zero, this indicates that input i has no request
to output j in subscheduler k .

§ 4.2 FUNCTIONS THAT MAY BE PERFORMED

A first aspect of the present invention may
function to relax the time needed for scheduling by
25 maximal-sized matching algorithms. (Recall that with prior
techniques, the maximal matching has to be completed within
one cell time slot.) The present invention may accomplish
this by operating several subschedulers in a pipelined
manner. In this way each cell may use more than one cell
30 time slot to reach arbitration, so long as the

subSchedulers, collectively, finish matching for at least one cell every time slot. The present invention may insure that this occurs by providing K subSchedulers, where each of the K subSchedulers generate a match in no more than K cell time slots.

The present invention may also function to provide high throughput and fairness in best-effort traffic. The present invention may accomplish this by adopting an appropriate maximal matching algorithm, such as iSLIP or DRRM for example. Either algorithm may be implemented because the present invention may provide a match every time slot. These algorithms have been proven to provide 100% throughput under uniform traffic, and maintain fairness for best-effort traffic.

§ 4.3 EXEMPLARY OPERATIONS

Figure 2 is a bubble chart illustrating operations that may be performed in accordance with the present invention. When a new cell arrives at the switch, the VOQ operation(s) 215 may place the cell in the appropriate VOQ 115'. It 215 may also inform the main scheduler operation(s) 220 about the stored cell. Next, the main scheduler operation(s) 220 may increment the value $(C(i,j))$ of the request counter $(RC(i,j))$ corresponding to the VOQ $(VOQ(i,j))$ storing the cell. That is, $C(i,j) = C(i,j) + 1$.

At the beginning of every cell time slot t , the main scheduler operation(s) 220 may check to see if there are any cells in a VOQ that are waiting dispatch

arbitration. If it finds that there is a cell waiting for arbitration, and that the next subscheduler is not already handling a cell in the same VOQ (i.e., if $C(i,j) > 0$ and $F(i,j,k) = 0$, where $k = t \bmod K$), it may decrement the value (C) of the associated request counter (RC) 222 and set the associated request flag (RF) 224 (i.e., $C(i,j) = C(i,j) - 1$ and $F(i,j,k) = 1$). If the foregoing conditions are not met (i.e., if $C(i,j) = 0$, or if $F(i,j,k) = 1$) it may leave the value (C) of the request counter (RC) 222 and the value (F) of the request flag (RF) 224 unchanged.

Subscheduler k 228 may perform a maximal-sized matching operation 226. It may do so at a time t, where $Kl + k \leq t < K(l+1) + k$, where l is an integer. In one embodiment, to apply the DRRM maximal matching algorithm as a matching algorithm in a subscheduler k, $F(i,j,k)$ may be used instead of VOQ requests as occurs in DRRM. Each subscheduler k may have its own round-robin pointers. The position of pointers in a subscheduler k may be modified by the results only from subscheduler k. The operation of DRRM in a subscheduler k may be the same as that of the non-pipelined DRRM scheme.

By the end of every time slot t, the subscheduler k 228 that started the matching operation K time slots previously (i.e., where $k = (t - (K-1)) \bmod K$), will have completed the matching operation 226. When an input-output pair (i,j) is matched, the main scheduler operation(s) 220 may unset the k^{th} subscheduler's corresponding flag (i.e., $F(i,j,k) = 0$). In this case, the VOQ operations 215 may be allowed to send the HOL cell in the associated VOQ 115' to

output at the next time slot. This may be used to ensure that cells from the same VOQ 115' are transmitted in sequence, even if $L(i,j) - C(i,j) > 1$, where $L(i,j)$ is the occupancy of VOQ(i,j) (Note that $L(i,j) - C(i,j) =$

5 $\sum_{k=0}^{K-1} F(i,j,k)$). When an input-output pair is not matched, the flag is not changed. Note that in one alternative, the subscheduler operation(s) 226 may reset the flag ($F(i,j,k)$) directly.

10 Timing operations 250 may determine a current cell time slot (t), and using parameter K 255, may determine control signals based on its present determined value k.

15 Having described various operations that may be performed in accordance with the present invention, exemplary apparatus, methods and data structures for implementing such operations are now described in § 4.4 below.

20

§ 4.4 EXEMPLARY METHODS AND APPARATUS FOR PERFORMING THE EXEMPLARY OPERATIONS

25 Exemplary methods that may be used to implement various operations of the present invention are described in § 4.4.1. Then, exemplary apparatus that may be used to implement various operations of the present invention are described in § 4.4.2.

30

§ 4.4.1 EXEMPLARY METHODS

Figure 3 is a high-level flow diagram of an exemplary method 215' that may be used to effect a virtual output queue operation 215. The order in which the steps are effected is not intended to be limited to the order shown. Referring to conditional branch point 310, when a new cell arrives at the VOQ(i,j) 115', the main scheduler operation(s) 220 may be informed, as indicated by block 320. (See, e.g., communication 230 of Figure 2.) Referring to conditional branch point 330, if a previous match request has succeeded, the HOL cell in VOQ(i,j) 115 may be sent to output j (e.g., in the next cell time slot). A successful match request indication may be provided directly from the subscheduler (k) (See, e.g., communication 240 of Figure 2.), or via the main scheduler operation(s) 220.

The main scheduler operation(s) 220 serve to manage the request counters 222, the request flags 224, and the subscheduler operations 226. The acts related to these management responsibilities may be triggered by events (e.g., the entry of a new cell into a VOQ) and may be run periodically or in accordance with some timing-based control (e.g., based on a value (k) derived from a current time slot (t)). Indeed, the timing operations 250 of Figure 2 may be thought of as one of the main scheduler operation(s) 220. Figure 4A is a flow diagram of an exemplary method 220a' for event-triggered acts of the main scheduler operation(s). Figure 4B is a flow diagram of an exemplary method 220b' for timing-controlled acts of the main scheduler operation(s). Each is described below.

Figure 4A is a high-level flow diagram of an exemplary method 220a' for effecting main scheduler event-triggered operations. Referring to the conditional branch point 410 and block 420, when a new cell arrives at a VOQ, this fact is relayed to the main scheduling operation(s) 220 (Recall 310 and 320 of Figure 3.) and the count value $C(i,j)$ of the counter $RC(i,j)$ corresponding to the VOQ receiving the cell, is incremented.

Figure 4B is a high-level flow diagram of an exemplary method 220b' for effecting main scheduler timing-controlled operations. The order in which the steps are effected is not intended to be limited to the order shown. As shown by loop 440-490, the main steps of the method 220b' are run for each cell time slot (t) (e.g., at the start of each cell time slot). As shown by loop 450-480, the main steps of the method 220b' may be run for each VOQ. Note that separate instances of the method 220b' could be run in parallel for the various VOQs. As shown in block 460, k is set to $t \bmod K$. (Note that k can be thought of as "timing-based".) Referring now to conditional branch point 470, the method 220b' may determine whether $C(i,j) > 0$ and $F(i,j,k) = 0$. That is, the method 220b' may determine if there is a cell in $VOQ(i,j)$ awaiting dispatch arbitration and if subscheduler K is available to $VOQ(i,j)$. If these conditions are met, the k^{th} subscheduler 228 may be informed, e.g., by setting flag $F(i,j,k)$ to "1", as indicated by block 472. Further, the request counter may be decremented ($C(i,j) = C(i,j) - 1$), as indicated by block 474.

Figure 5 is a high-level flow diagram of an exemplary method 226' that may be used to effect each of the subscheduler operation(s) 226. As indicated by conditional branch points 510 and 520, and block 530, in each case, the particular subscheduler k may run a matching operation when the time slot $t \geq K_l + k$ and $t < K_{(l+1)k}$. (This is illustrated for $K = 3$ in Figure 8, described in § 4.5 below.) That is, as indicated by conditional branch point 520, it is determined whether or not the request flag (RF) is set to one ($F(i,j,k) = 1$). If so, the subscheduler k may run the matching algorithm for the (HOL) cell of $VOQ(i,j)$ as indicated by block 530. Further, at the end of the time slot, a subscheduler will have finished its matching operations. This will be the case for subscheduler k , where $k = (t - (K-1) \bmod K)$. Thus, as indicated by conditional branch points 540 and 550, if $k = (t - (K-1) \bmod K)$, it is determined whether or not the match request was successful (for $VOQ(i,j)$). If so, the request flag $RF(i,j,k)$ may be unset ($F(i,j,k) = \text{zero}$) as indicated by block 560, and the appropriate VOQ may be informed of this, as indicated by block 560, so that it will send its (HOL) cell to the appropriate output. Note that in this embodiment, each subscheduler k generates a match for each requested output port. An output port is requested if any of the $VOQs(_,j)$ is non-empty.

§ 4.4.2 EXEMPLARY APPARATUS

Figure 6 is a high-level block diagram illustrating exemplary components and interconnections of

an exemplary apparatus 600 that may be used to effect at least some of the operations of the present invention. Each of the inputs 610 includes (or more generally, is associated with a corresponding) a number of virtual output queues (VOQs) 615 corresponding to the number of outputs. An exemplary main scheduling module 620 may include (or more generally, be associated with) K subschedulers 650 and N^2 request counters (RCs) 640, each of the request counters (RCs) being associated with a VOQ (and therefore an input-output pair). To simplify the drawing, only the components associated with the VOQs of one of the inputs are shown. Each of the K subschedulers 650 may include (or more generally, be associated with) N^2 request flags (RFs) 660, each of the request flags (RFs) 660 in each of the subschedulers being associated with a VOQ (and therefore an input-output pair). The VOQ operations 215 may be effected on the virtual output queues 615. The main scheduler operations 220 and timing operations 250 may be effected on the scheduling control module 630. Finally, the subscheduling operations 226 may be effected on the subschedulers 650. Naturally, such operations only need to be associated with the VOQs of an input, and can be effected in many other ways.

There are various signaling lines and links that may be provided. The solid lines may be used to indicate the receipt of a cell, and the transmission of the cell to the output via the (e.g., crossbar) switching fabric 680. The signaling line(s), depicted by the dashed line 692, may be used to indicate, to the scheduling control module 630, the arrival of new cells, and to indicate to the VOQs when a match request was successful. The signaling lines,

depicted by dotted lines 694 and 696, may be used by the scheduling control module 630 to control the request counters 640 and the request flag 660 values, and to permit communications between the scheduling control module 630 and each of its subschedulers 650.

The VOQ operations 215, the main scheduler operations 220, and the subscheduler operations 226 may be effected by programmable logic arrays, integrated circuits such as application specific integrated circuits, and/or microprocessors operating in accordance with (e.g., stored) instructions. Memory (referred to generally as a "machine readable medium") may be used to store the various state information, i.e., the values of the request counters and request flags), used by these components. Similarly, memory can be used to buffer cells at the virtual output queues.

§ 4.5 EXAMPLES ILLUSTRATING OPERATIONS PERFORMED BY AN EXEMPLARY EMBODIMENT

Figures 7(a) through 7(c) illustrate an example of operations of a pipelined maximal-sized matching scheduling method employing subschedulers in accordance with one embodiment of the invention. In this example, the invention is embodied in an input-buffered switch, such as that illustrated in Figure 1. In the exemplary switch, $N = 10$ and $K = 3$. The request counters (RCs) and the request flags (RFs) are initialized, i.e., $C(i,j) = 0$ and $F(i,j,k) = 0$.

Two previous cells have arrived and are being matched by subscheduler 0 and subscheduler 1. In Figure 7a, these cells can be seen waiting in VOQ (9,0) and VOQ (9,9), respectively. As indicated, the request flags of RF(9,0,0) and RF(9,9,1) have been set to "1". Also indicated, the request counters RC(9,0) and RC(9,9) have been incremented upon arrival of the cells at their VOQs, and decremented upon assignment to the subschedulers.

Still referring to Figure 7a, in time slot $t = 1$ a new cell arrives in input 0. The cell is to leave the switch on output port 0. Accordingly, it will be stored in VOQ(0,0). Figure 7a shows the cell stored in VOQ(0,0). The main scheduler will become informed of this and will increment the request counter RC(0,0) corresponding to VOQ(0,0), ($C(0,0) = C(0,0) + 1$). (Recall, e.g., Figure 4A.)

Referring to Figure 7b, at the beginning of time slot $t = 2$, when $k = 2$ ($= 2 \bmod 3$), the scheduling control module will read RC(0,0) and see that C(0,0) is equal to one, and it will read RF(0,0,2) and see that F(0,0,2) is equal to zero. (Recall, e.g., 470 of Figure 4b.) Dashed lines represent these actions. RC(0,0) will be decremented, $C(0,0) = C(0,0) - 1$, and RF(0,0,2) will be set to one. (Recall, e.g., 474 and 472 of Figure 4b.) Solid lines represent these actions. Within this cell time slot, subscheduler 2 begins to operate the maximal-sized matching according to the adopted algorithm. (Recall, e.g., 510, 520 and 530 of Figure 5.) This is because $2 \geq 2$ ($2 \geq K_1 + k = 3(0) + 2$) and $2 < 5$ ($2 < K(l+1) + k = 3(1) + 2$).

Subscheduler 2 will have completed its matching operation by the end of the fifth time slot ($t = 5$).

Referring to Figure 7c, at the end of time slot $t = 3$, when $k = 1$ ($= (3 - (3 - 1)) \bmod 3$), the $k = 1$ subscheduler will have completed its matching operation. Recall that earlier cell arrived in input port 9, at time slot $t = 0$, and is to leave on output port 9. If the input-output pair (9,9) is matched, the flag $RF(9,9,1)$ will be unset (i.e., $F(9,9,1) = 0$), and the $VOQ(9,9)$ will be informed that it won arbitration and that it can send its head of line (HOL) cell to output 0 (e.g., in the next time slot). (Recall, e.g., 550, 560, and 570 of Figure 5.) Similarly, as indicated by Figure 8, at the end of cell time slot $t = 4$, subscheduler 2 will finish matching, and at the end of cell time slot $t = 5$, subscheduler 0 will finish matching. If new cells continue to arrive, this pattern will continue and the main scheduler will be able to produce a match at the end of every cell time slot.

Therefore, as this example illustrates, the main scheduler was given more time to complete the matching by allowing three time slots instead of just one. In addition, the subschedulers may implement maximal matching algorithms that provide 100% throughput under uniform traffic, and maintains fairness for best-effort traffic.

§ 4.6 CONCLUSIONS

In view of the foregoing, the scheduling technique and apparatus of the present invention relax the time for scheduling, thereby enabling large scale and/or

high throughput switching, while maintaining fairness for best-effort traffic. Other approaches, such as WRRGS, may fail for best-effort traffic. For example, if the allowable arbitration time per iteration, a port speed, a cell size, and the number of iterations are T , C , L , and I , respectively, then, $T = (KL)/(CI)$, where K is the number of subschedulers. In the non-pipelined DRRM scheme, K will essentially be equal one. The foregoing equation shows that by adding subschedulers, the time for arbitration will increase. For example if the time for arbitration for a non-pipelined DRRM scheme was 3.2 ns, that time can be increased to 9.6 ns by implementing three subschedulers. Having more time for arbitration may become important, because it is difficult to implement round-robin arbitration that supports large numbers of inputs in hardware using available CMOS technology, in which, for example, typical gate-delay time is about 100ps. Refer to Texas Instruments, "GS40 0:15-micrometer CMOS, Standard Cell/Gate Array," <http://www.ti.com/>, version 0.2, May 2000. Incorporated herein by reference. Therefore, the present invention achieves the desired number iterations even when the number of inputs increases or the port speed increases.

To reiterate, the present invention can achieve 100% throughput under uniform traffic, and maintain fairness for best-effort traffic. The known maximal-sized matching algorithms that meet both of these conditions require that a match be made at the end of every time slot. This becomes difficult, if not impossible, with decreasing arbitration times. The present invention advantageously allows for longer arbitration times, while still producing

a match at the end of every time slot.

Given these conditions the present invention may allow the increase of inputs and output ports, and/or may
5 allow for the increase in port speeds. In view of the foregoing, the present invention can provide a scalable switch that has high throughput and maintains fairness for best-effort traffic. The present invention can avoid the problems of (i) the increasing arbitration time as switch
10 size increases, and/or (ii) decreasing cell time slots as port speed increases, by permitting the matching algorithm to use more time.